ISSN 0112-3351

BEEBLET

CONTENTS

Editorial Notes 2
A Glossary of Computer Language 2
Watford Sideways RAM board 4
Something Bigger 7
HELP 13
Getting Away From Games 15
A Computer In The Kitchen? 16
AGM Minutes 22
Electron Column? 26
It Takes All Sorts 27
Advanced Disc Filing System 32
Plinth 34
Civilised Mode Change 35
Budget Memory Expander 37
Classified Ad 37
Letters.. 38
Hall of Fame 39
Local Groups 40

VOLUME 4 NUMBER 3 APRIL 1986

BBC / ACORN

COMPUTER USER GROUP NZ (inc)

EDITORIAL NOTES

Last months mag was rather latter than I expected - the delay on my part got multiplied by the printers backlog before Easter and some confusion regarding delivery. I am creeping back onto schedule.

Unacknowledged Contributions.

I apologise to those whose contributions I have published but not labelled. The reason for this is that there name was not on their contributions, which tend to get sorted into appropriate piles and processed very rapidly. Covering letters are not always passed on to me, especially if they have material for some other committee member. The cartoons in this issue were kindly created by Ricardo De Lee, while Anne F. is the author of 'something bigger' and a couple of other contributions this issue (pick her style?).

Hall of Fame

As a number of recent correspondents have noted, this hasn't changed much for ages. Is no one playing games any more? Anyway, next month any scores which have been posted for more than six months will be wiped. Any new entries will be welcome.

Beeblet is Edited by Bruce Wills who can sometimes be found on 861-662 (home) or 729-929 x402 (work).

DISCLAIMER

The views of the reviewers, the Editor and other contributors are not necessarily shared by the BBC/ACORN COMPUTER USER GROUP NZ (INC).

A GLOSSARY of COMPUTER LANGUAGE

Source: Systems Resources Ltd (unwittingly)

ACCESS TIME - The time between the instant at which information is called for and the instant at which management expects the final report. These "instants" occur between one tick of clock and the next.

ANALYST - Anyone who sd designates themselves.

ANALOG COMPUTER- Machine to measure the timber in a virgin forest.

ARITHMETIC SHIFT - Preferred apparel for female programmers.

ASSUMED DECIMAL POINT - Located two positions to the right of programmer's estimate of their own worth.

B-BOX - A hive

BLOCK DIAGRAM - Schematic gibberish.

CHARACTER - omeone interested in computers (see also Special Character)

CHAINING - method of attaching programmers to desks to speed up output.

COBOL - Two people having fun.

CODE - A respiratory ailment

COMMON LANGUAGE - The first thing a programmer must forget to be successful.

CRYOGENIC FILM - A sad movie on hygiene.

CYCLE - Half a bicycle.

DATA FLOW DIAGRAM - Unstable block diagram.

DANGER: 240,000 MILLIVOLT SOURCE - Standard 240 volt wall plug.

DOUBLE PRECISION - Ten fingers on each hand.

ERROR - what someone else has made when they disagree with your computer output.

EXTERNAL STORAGE - Wastebasket.

FIXED WORD LENGTH - Four letter words used by programmers when in a state of confusion.

INDEX REGISTER - Constant source of irritation.

KEY PUNCH OPERATOR - Best informed sources regarding weaknesses in the system, new large scale computers, and the best bars in town.

LATENCY - Normal arrival at work.

LOW ORDER POSITION - Programmers position in the chain of command.

MACRO - The last half of an expression of surprise : as in "Holy macro!".

MAINTENANCE MANUAL - A Mexican Handyman (see IBM).

MATHEMATICAL MODEL - 42-24-38

OPTICAL SCANNER - censored

PROGRAM IMPLEMENTATION - Hire more people and expand the office.

PULSE TRANSFORMER - censored

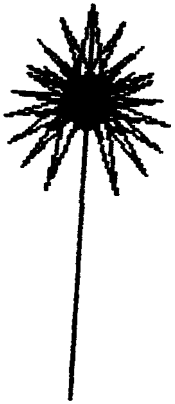
RANDOM ACCESS - Promiscuous

UNDER CONSIDERATION - Never heard of it.

UNDER ACTIVE CONSIDERATION - We're looking in the files for it.

VARIABLE WORD LENGTH - Stammering.

ZERO SUPPRESSION - Absolute freedom.



Watford Sideways Ram Board

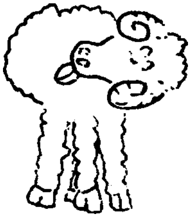
Richard Jack, Nelson

A short reply to a *H in the Christmas Beeblet regarding the Watford MK.2 Rom/Ram board which came without any operating instruction.

Yes,there was a review in Acorn User, just a month after we had received our board. The reviewer gave me a strong impression that he did not like it and gave it eleventh out of twelve boards compared. You wouldn't buy one after reading that review but when I re-read "the criteria for choosing an expansion board " in the April issue I also read that amongst the major problems of fitting expansion boards was "when fitted with 12 or more devices it becomes quite heavy ,leading to difficulties if the computer is moved...." I was relieved he had really not been serious at all .The board is well and settled in... write protect and battery-backup switches had to be fitted to get round certain situations but all appears to be well and the board will run any of the rom programs tried to date.

Maybe our Ed. could find someone able to give suitable definitions to some of these terms (and others) often associated with sideways ram -

cmos
dynamic
fast
shadow
static etc.



Ed's Notes:

I am not sure exactly what you are interested in discussing in relation to sideways RAM, however the above terms fall into two broad areas-
1/ the physical electronic components and their characteristics
2/ the nature and operation of sideways RAM.

Electronic components:

There are a growing number of electronic devices, in the form of silicon "chips", associated with computer memory. These can be broadly divided into two groups: those that the computer processor can only read from (like a record player "reads" records), and those that it can both write (ie. save information) to, and read the information back from.

The first group comprise:
-ROMs (Read Only Memory), normally supplied by manufacturers or software houses with programs etc indelibly etched in them;
-PROMs (Programable Read Only Memory), which are ROMS which can be loaded with data (programed) by either manufacturers or hobbists;
-EPROMS (Electrically Programable Read Only Memory) which can redily be programed (given the appropriate hardware) and subsequently erased (using ultraviolet light through a window in the chip) and re-programed;
-EEPROMS (Electrically Erasable Programable Read Only Memory).

The list goes on.

The devices of interest in the second group comprise various types of RAM (Random Access Memory). RAM can be thought of as a large array of cells, each of which can be individually addressed by the computer processor (ie any cell may be addressed at random) to read or write information. RAM is so designated to distinguish it from serial devices (such as shift registers) which comprise a sequence of cells, into which data can only enter or leave though an end - like cassette tapes : you have to wind them through until you reach the data you want. The RAM label is misleading - you generally have "random access" ROMs also!

Memory can also be classified into volatile (turn off the power and it forgets all the information stored in it) and non-volatile. ROM falls into the latter category: RAM is normally volatile.

RAM chips have various characteristics in terms of power consumption and speed depending on the materials (generially mainly silicon) and method of manufacture. CMOS chips for instance generally require very little power when they not being actively addressed (for reading or writing), and are thus suitable for battery backup so your don't lose the data they are storing. They also have other desirable characteristics as regards interference from other devices (useful eg. in an industrial situation). CMOS RAM is however more expensive than equivalent speed ordinary RAM.

RAM speed relates to how quickly the memory devices can respond to requests from the computer processor to store or retreive information. Typical speeds range from 150 to 650 ns. It is relevant to the extent that the faster the processor, the shorter the time available for the memory to respond if the processor is not to be held up waiting for the data it wants to be transfered to or from it.

"Static" and "Dynamic" RAM chips are also available. The latter is of simpler construction and cheaper, but has to be periodically "refreshed" - ie each cell has to be read every few moments otherwise it will lose its data. Both are volatile ie need the power on all the time to retain their data. The type (static/dynamic) of RAM memory makes little practical difference unless you intend to build your own memory boards.

Memory organisation

"Shadow memory" means much the same thing as "sideways" RAM. The Beeb's 6502 processor is capable of addressing what can be thought of as a matrix containing some 65,000 ("64K") cells, each of which can store eight binary digits (or two alphanumeric characters). This is divided up into blocks as illustrated on page 500 of the UG. Some 16K (from hexidecimal address 8000 up to BFFF, or the area between the top of screen memory to the bottom of the Operating System) is marked "4 paged ROMs eg BASIC". This reference is to the 4 sockets in the Beeb into which you can plug ROMs (includes EPROMs etc), and one of which is normally occupied by the BASIC ROM. Now, each of these ROMs occupy the same 16K block of memory (ie they sit sideways from this section of the memory map) - but only one is addressed, or paged in, at the time. It is relatively simple to switch, or page, blocks of RAM into (and out of) this space as required - thus sideways RAM. Shadow RAM refers to RAM which can be switched into the address space normally occupied by ROM.

SOMETHING BIGGER:

Writing longer programs

Pt I

Comes the time you want to write a program that does more than can be handled by a couple of PROCs. This series will take you through a fairly straightforward game and give you some idea of just how much work is really involved, as well as offering some techniques for your own adaptation.

THE GAME:

Certain people who ought to remain nameless have used the computer so much for playing Snapper, and been playing it so badly, that Pacman has got out of the maze and into the maze of RAM chips inside your computer, and is now busy gobbling up bytes of its memory instead of dots on the screen. He is "amenable" to guidance however, and the player's task is to lead him out with the minimum loss of memory.

GAME STRUCTURE:

Games appear to have three parts to them:

1. Instructions
2. the Game itself
3. Some way of ending, with high scores etc.

From the programmer's point of view however, there are several more stages, invisible to the player:

1. Set up all the shapes (user-defined characters), choose the colours, set up variables needed by the program.
2. Print out the instructions for the player, allow enough time to be read, and provide some way of moving on...
3. Set up the screen picture board to play on.
4. Get and analyse voves made by the player
5. Action these moves on screen and within the program
6. Update the scoreboard
7. At the end, produce the "High Score" or whatever the equivalent will be.
8. Clean up ie return colours, Key actions etc to normal.

This month we'll deal with points 1-3; or as the player sees it, part 1. The program has been written to be as machine-independant as possible, and easily convertible to other screen modes too. There is one quirk which we don't meet this month (thank goodness!!)

THE LISTING

Line 10 makes it easy for you to track down errors when they occur within the program; at the end of the series it will be rewritten slightly to be more useful.

```
10 ON ERROR REPORT:PRINT" @ line ";ERL:STOP
```

Lines 20-70 are comments, for future reference. As you can see, it was written almost a year ago for another reason.

Lines 80-180 are what programmers love to call "the main line" - that is, it begins at the beginning and goes to the end supplying bare bones only. It looks more like the player's idea of the program than the programmer's, but it gives the casual 'reader' an idea of what the program goes through.

PROCset_up

In the game, this version of an Acorn computer (the Neutrino?) has got 32K of RAM made up of RAM chips holding 1, 2, 3, or 4 bytes of memory in each. (1024 bytes makes 1k, so this 32K has 32,768 bytes). A little impractical, but that's poetic licence for you! Anyway, we will need to define shapes to represent each of the possible 4 RAM chips used. Here characters 225-228 are used. I always define characters using hexadecimal for reasons stated in previous issues of 'Beeblet'.

Pacman can move in 4 directions, so we'll need 4 shapes to represent which way he is going (CHR$229, 231, 233, 235), plus his mouth (230,232,234,236). This is because his mouth is to be a different colour to the rest of him.

Something is needed to show the chips eaten out by Pacman (237), and some way to show the exit the player is aiming for (224)

COLOURS:

Mode 1 offers you the choice of 4: red, yellow, white and black. Instead of a yellow Pacman, how about a blue one, and to show his mouth opens and shuts, let's cheat and use a flashing blue/yellow, so when it is shut it is blue like the rest of him, and yellow when it's open, all with no further effort on the programmer's part. Lines 350 and 360 cope with this change to the actual colours displayed:

```
350 VDU19,1,11,0,0,0
```

Line 370 switches off the cursor on a Model B BBC; I'm not sure how the Electron would do it. Might be in its User Guide.

```
370 VDU23;11,0;0;0;0
```

CO-ORDINATES

Throughout the game, we're not using graphics co-ordinates to move round the screen, but text ones as used by PRINT TAB(X,Y), and X% and Y% will be used to represent where Pacman is at the present time (whenever that will be). So to start the game he will always be at the top column, and down 2 rows from the top of the screen.

```
380 X% = 0 : Y% = 2
```

We will also want to know where he was just before the player moved him, to know where to place an "eaten chip" shape. This is the "old X" and "old Y", or OX%, OY%. At the start of a game, the old position is the same as the current position, so it can be pre-set here too:

```
390 OX% = X% : OY% = Y%
```

OTHER:

Because we will be keeping track of how much memory Pacman has eaten up, we'll need a variable, "memory%", set up - initially at zero because he doesn't start eating until the player starts playing!!

```
400 memory% = 0
```

KEYS USED TO PLAY:

A lot of games use such keys as "U" for Up, "D" for down etc, or "A" up "Z" down "<" left ">" right or some other variation. We're going to use the cursor keys for the player to manoeuvre Pacman. To do this easily, *FX4,1 makes these keys return an ASCII value along the same lines as the other keys on the keyboard.

```
410 *FX4,1
```

PROCinstructions

First of all, simulate the screen of a computer that's just been switched on. Pause to allow the player to assimilate this (after all, he just typed in on the screen "CHAIN PACBEEB", and now expects something to happen), then throw a spanner in the works: "Bug Error" and "Bad Program".

```
510 wait = INKEY(100)
```

"wait" is a variable. The computer is being told to wait for 100 hundredths of a second (1 second here) before carrying on with its next instruction (this method also allows the player to hurry through by pressing a key rather than wait a full second). There is a different way of waiting used at line 730 also using INKEY:

```
730 REPEAT UNTIL INKEY(-99)
```

In this instance, the computer has to wait until the space bar has been pressed. The value it sends (as listed in the User Guide), is -99, so the computer keeps looking at the keyboard

for this Key and this Key only. It's an elementary reading test for the player?
INKEY with a positive number will wait the specified length of time OR until any Key is pressed; INKEY with a negative number will wait until the Key returning that value is pressed, and no other. (Or alternatively, until you turn the computer off - a little drastic).

The rest of this PROC deals with setting out the instructions on screen, including how to print 'unprintable' characters (those not on the Keyboard), and how to insert colour in the text.

```
 10 ON ERROR REPORT:PRINT" @ line ";ERL:STOP
 20 REM  Lost Memory
 30 REM  Pacman got out of his program!
 40 REM  BBC/Acorn Computer User Group NZ Inc
 50 REM  ***    Broadcast on Access Radio ***
 60 REM another hot-house project May '85
 70
 80 MODE1
 90 PROCset_up
100 PROCinstructions
110 CLS
120 PROCfill_screen
130 REPEAT
140   PROCget_move
150   PRINTTAB(5,1)"He has eaten ";memory%;" bytes"
160   UNTIL X% = 39 AND Y% = 30
170 PROCfinal
180 END
190
200 DEF PROCset_up
210 VDU23,224,&7E,&7E,&7E,&7A,&7E,&7E,&7E,&7E
220 VDU23,225,&20,&10,&88,&44,&22,&11,&08,&04
230 VDU23,226,&49,&92,&24,&49,&92,&24,&49,&92
240 VDU23,227,&55,&AA,&55,&AA,&55,&AA,&55,&AA
250 VDU23,228,&FF,&FF,&FF,&FF,&FF,&FF,&FF,&FF
260 VDU23,229,&1C,&3E,&6F,&7F,&1F,&0F,&3E,&1C
270 VDU23,230,&00,&00,&00,&40,&60,&70,&00,&00
280 VDU23,231,&38,&7C,&F6,&FE,&FC,&FC,&7C,&38
290 VDU23,232,&00,&00,&00,&02,&06,&0E,&00,&00
300 VDU23,233,&3C,&7E,&FF,&FF,&DB,&4E,&04,&00
310 VDU23,234,&00,&00,&00,&00,&30,&30,&38,&00
320 VDU23,235,&00,&18,&7E,&DB,&FF,&FF,&7E,&3C
330 VDU23,236,&3C,&18,&18,&00,&00,&00,&00,&00
340 VDU23,237,&00,&00,&7E,&42,&42,&42,&7E,&00
350 VDU19,1,11,0,0,0
```

```
360 VDU19,2,4,0,0,0
370 VDU23;11,0;0;0;0
380 X% = 0: Y%= 2
390 OX% = X%: OY% = Y%
400 memory% = 0
410 *FX4,1
420 ENDPROC
430
440 DEF PROCinstructions
450 PRINT"BBC Computer 32K"
460 PRINT'"BASIC"
470 PRINT'">RUN"
480 wait=INKEY(200)
490 PRINT'"Bug Error"
500 PRINT'"Bad Program"
510 wait=INKEY(100)
520 PRINTTAB(0,30)"Press SPACE BAR to HELP"
530 REPEAT UNTIL INKEY(-99)
540 CLS
550 PRINTSPC15"HELP  ME!"
560 wait=INKEY(100)
570 PRINT'"PACMAN has escaped from his program,"
580 PRINT'"and is now gobbling up all my memory."
590 wait=INKEY(200)
600 PRINT''"Please push him out."
610 PRINT'"I will show you a memory map, and you"
620 PRINT'"must move him out the door  ";
630 COLOUR2:PRINTCHR$224:COLOUR3
640 PRINT'"but make sure he eats as little more"
650 PRINT'"of my memory as possible."
660 wait=INKEY(200)
670 PRINT''SPC10;CHR$225;" is 1 byte of memory"
680 PRINT'SPC10;CHR$226;" is 2 bytes of memory"
690 PRINT'SPC10;CHR$227;" is 3 bytes of memory"
700 PRINT'SPC10;CHR$228;" is 4 bytes of memory"
710 PRINT''"Use the cursor keys to manoeuvre him."
720 PRINTTAB(0,31)"Press SPACE BAR to start";
730 REPEAT UNTIL INKEY(-99)
740 ENDPROC
750
760 DEF PROCfill_screen
770 LOCAL X%,Y%
780 FOR Y%=2 TO 30
790   FOR X%=0 TO 39
800     VDU31,X%,Y%,(224+RND(4))
810     NEXT X%
820   NEXT Y%
830 VDU31,39,30,17,2,224
```

↓↓↓

```
840 VDU31,0,2,17,2,231
850 VDU5:MOVE0*32,1023-2*32:VDU18,0,1,5,232,4
860 ENDPROC
```

PROCfill_screen

Just to tantalise you, here is the "board" the player will be using.

We are already using "X%" and "Y%" in the program, but line 770's use of LOCAL in effect sets up a different variable which will be used differently within this PROC only. That is, whatever is done to the X% and Y% of this PROC has no bearing on what any X% and Y% not called by this PROC may be doing. Of course, there is no need to re-use the same variable name in this way for this program, but in longer ones with lots of variables it may be necessary if only to conserve memory to run that program in.

We want the screen TAB lines 2-30, and TAB columns 0-39 to represent the board, and therefore to be filled with shapes representing the RAM chips Pacman will be chewing through. The filling of the board is done within 2 "nested FOR...NEXT loops" ie one loop is nested within an outer loop - for every line (Y%), go across every column (X%=0-39) and put a character in that position.

VDU 31 is exactly the same as PRINTTAB.

```
800 VDU31,X%,Y%,(224+RND(4))
```

Line 800 could equally well have been written as:

```
800 PRINTTAB(X%,Y%)CHR$(224+RND(4))
```

The chip shapes were defined as characters 225-228, so by adding a random number to 224, we will guarantee that one of these 4 characters only will be used.

The next 2 uses of VDU31 also show the way of getting colour as well as a user-defined character printed out:

```
840 VDU31,0,2,17,2,231
```

could have been written as

```
840 COLOUR 2:PRINTTAB(0,2)CHR$231
```

ie VDU17 means "change text colour" (as opposed to graphics colour), "to the colour of the next number" in this instance, to colour 2 (which at line 360 we defined as blue).

If you type in the part of the program listed above, the program should run until it gets to line 140.

Next month: points 4-6 - playing the game (along with the rest of the program lines).

↓↓

```
**************************************************************
**************************************************************
*****  ******  ***          ***  ********       ****  ********
*****  ******  ***  ***********  ********  ***** ***  ********
*****  ******  ***  ***********  ********  ***** ***  ********
*****          ***       ******  ********       ****  ********
*****  ******  ***  ***********  ********  *********  ********
*****  ******  ***  ***********  ********  *******************
*****  ******  ***          ***       ***  *********  ********
**************************************************************
***********                                     **************
*********** A forum orchestrated by Steven Rose **************
***********                                     **************
**************************************************************
```

This is the first of the 1986 Help columns, which this year I will be presiding over.

In case anyone's interested, I'm a professional programmer, and I'll be calling on other members for hardware help.

Well, obviously you all know everything about BBCs, because Bruce hasn't received anything for this column recently. I'm doing this at fairly short notice this time, so I'll just have to invent some problems, and you'll have to tell me via Bruce which ones I should have solved.

1. At the last Wellington User Group meeting, I was shown half a page of justified text printed from Wordwise. The last line of one middle paragraph was about twenty characters and five words long, but had been spread out over the whole line. Other short lines had been correctly left unjustified.

Does anyone have any idea why this could have happened?

ED'S NOTE: You all may have noticed that Beeblet is lined up down right and left edges (ie justified) - I get lots of these problems. Make sure that there are no hidden control codes at the end of the line (ie. delete everything up to the next sentence), then add at least one carriage return - another if you want to separate your paragraphs with blank lines.

All problems are really lack of knowledge, so here is some to add to your collection.

2. Did you know that *Build files can include comments?

→→

If you begin a line with '* ', it is ignored by the operating system, so you can put anything you like after it.

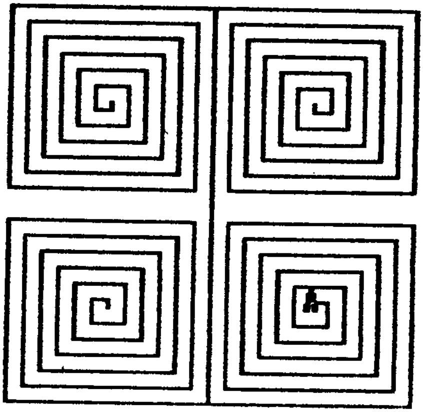
3. *Build files can be editted by any word processor or text editor that can produce a text-only file (that is, without control characters other than carriage return and line feed).

If you are using Wordwise and want to include comments (see 2), use the embedded command "PCx" where x is an unused character. This redefines the Pad Character (which produces a space when printed) from " " to x. I usually use # (hash) or ~ (tilde).

4. If your program is so long that every byte is important, try abbreviating reserved words when you type them in. The reason for this is that after the line has been entered, it is tokenised by BASIC (reserved words like PRINT are replaced with a single byte). The line therefore becomes shorter than when you typed it in. By abbreviating the reserved words, less bytes are thrown away during the tokenising process, so you fit more into the line, and reduce the number of lines, with their associated four byte overhead.

5. Finally, H E L P ! What would you like to see here next time ? Do you know all about BASIC? What don't you know about modems, assembly language, disk filing systems and choosing them, sideways RAM/ROM ?

Well, don't just sit there...



GETTING AWAY FROM GAMES

Anne F.

This was Fraser's problem in Auckland - how to do something useful, like solving equations, instead of adding to High scores. He even sent down example.

So apart from referring all you who want their computer to do maths (after S.C and UE???) to the User Guide Sample Programs (the one in particular that solves quadratic equations), here is a program to produce the answer to Fraser's initial offering.

THE LISTING

The essential lines are lines 90 - 140. The rest is "padding". Reference was made to Chapt 26 Operator Precedence to check on the order the Beeb does its arithmetic.

```
   10 REM to get Fraser away from games!!
   20 REM and into getting a Ph.D in Computerised Maths
   30 MODE6
   40 PRINT"The equation to solve is:"
   50 PRINT
   60 PRINTSPC4"1.5(x)(y)"
   70 PRINT"y = _________   {LOG(1+P/Q)}"   80 PRINTSPC6"2
240A"
   90 INPUT'"Value for x",x
  100 INPUT"Value for z",z
  110 INPUT"Value for A",A
  120 INPUT"Value for P",P
  130 INPUT"Value for Q",Q
  140 PRINT"y = ";1.5*x*z*LOG(1+P/Q)/(2240*A)
  150 END
```

The program could be made to loop round itself to allow you several opportunities to try out different variables. Add:

```
   85 REPEAT
  143 REPEAT
  145 INPUT'"Another set of Variables?",ans$
  147 UNTIL INSTR("YyNn",ans$)
  149 UNTIL ans$="Y" OR ans$="y"
```

The whole program could in turn be put in a larger loop which asks you to input the equation as well. Have fun.

COMPUTER IN THE KITCHEN?

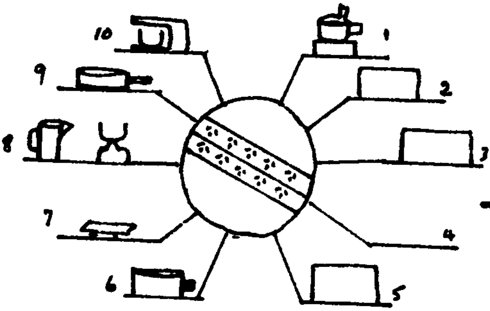
Sadie Ash

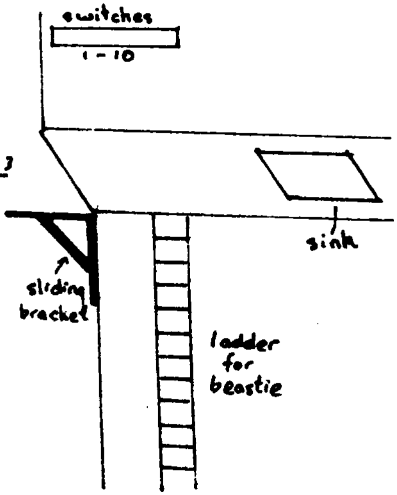
What a crazy idea - flour down the keyboard, fruit juice smeared over the VDU, toast in the disc drive... the quickest way to wreck a computer surely!!

However, with the setup I have acquired, I wouldn't be without it. The kitchen is very small, with limited bench space, and I was constantly bending down or stretching up to get the next appliance, trying to find temporary bench space for it and a power point within reach of its cable. To make matters worse, there was a totally inaccessible nook beyond a bit of bench that was actually quite a reasonable size. As the family grew and more food preparation was needed to cope with various allergies (both real and assumed), I got more and more frustrated with the situation. Whilst there was money for appliances, or hubby's hobby, or car repairs just before his weekend in Taupo fishing, something as major as a kitchen extension and remodelling was out of the question.

A few months back I took a leaf out of the fair ground (metaphorically), and built a ferris wheel in my kitchen! It centred in that hitherto useless nook, and thus was fairly economical with space that mattered. The idea was, that since seats on a ferris remained parallel to the ground while the wheel rotated, these 'seats' could accomodate kitchen appliances. (You thought I should have one in the front jungle to attract the neighbours' kids??) A diagram may be more use for you...

Key:
1 Food processor - outdoes Mac the Knife
2 toaster - 4-slice job
3 Microwave - tsunami-size
4 "reserved for future expansion"
5 Wotec grilla-oven - good for chops, chumps and chimps too
6 Crockpot
7 sandwich makers
8 jug and percolator - use only "Boggs filters & grounds
9 electric frypan - came with 5 year "never wash" guarantee
10 cake mixer - most used in conjunction with Capt. Hook's hand



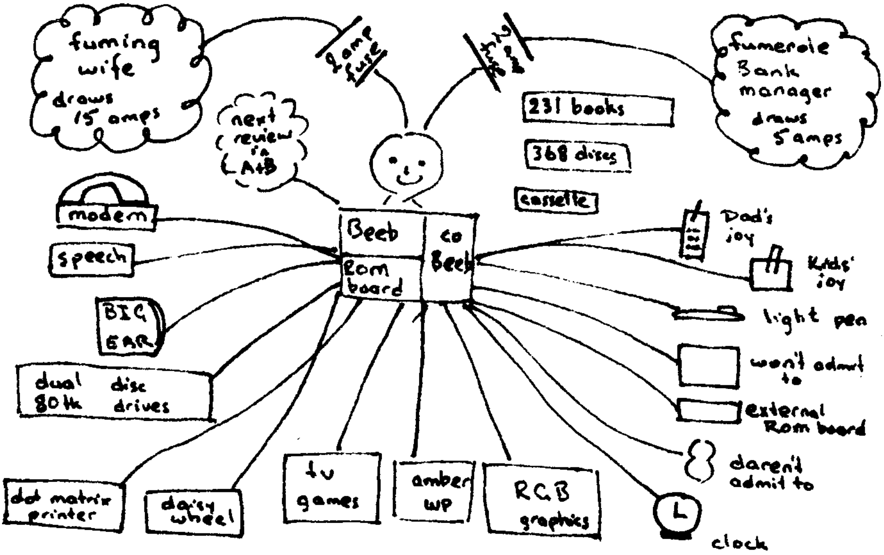


At the centre of the wheel, as an axle, is a multiplug box which all the appliances plug into, so they are 'on' all the time.It rotates round, so designing a cable from it to the mains that didn't twist as it rotated was a thrilling challenge for an electrician friend who prefers to remain anonymous. The axle now incorporates a motor, and there is a row of switches on a wall. Press a switch, and the wheel rotates to bring that numbered seat round to meet the main bench. A bracket slides out from the main bench to add further support while the appliance is in use (food processors and cake mixers can walk, and the microwave is heavy!)

HUBBY'S HOBBY

See diagram on next page...

Micro User Bob's wife Andrea had all my sympathy. "Beeblet's" Fischer-Technic advert in December looked like wrecking my Christmas budget, as well as cannibalising the kids' Lego. As hubby showed off his latest budget-wrecker though, this time I could see a faint glimmer of an application relevant to me. Previously, the only use the computer had for me was to get the Christmas newsletter done - I wrote it out once by hand, neatly, legibly, and spelt correctly, showed variations for different relatives, and handed it to hubby to produce the umpteen personalised copies he bragged the computer could do. He bribed one of the Kids. Kid claimed he needed 'Spellcheck' as well. He did too. Must have a word with his optometrist.

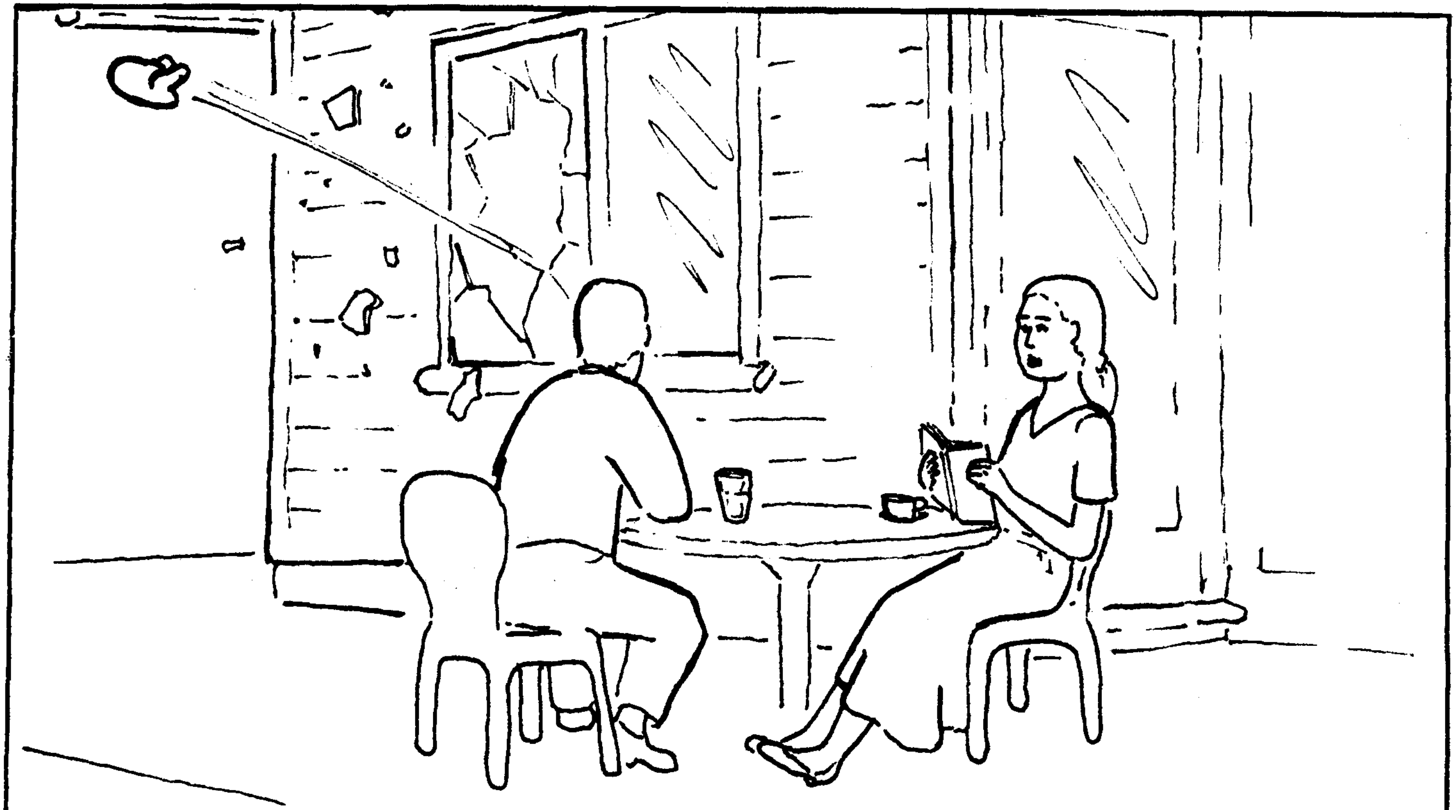


Anyway, now I get an extra 10 minutes in bed each morning, while the computer starts breakfast!! The night before, I fill the kettle, the coffee percolator, place 19 slices of bread in a pile beside the toaster, crack a dozen eggs into the food processor with a bottle of milk, and place a bowl of porridge mix in the microwave. Upon pain of no tv for a month, the last person to use the computer boots up an execution file that looks like this:

```
clock 0630
*beastie
ferris
wave simple 500
clock 0635
kettle
perc
proc 3 20
toast
end
```

You want an explanation?

clock 0630 when 'it is 6.30am, do the following:*beastie calls the beastie robot start-up routine (which

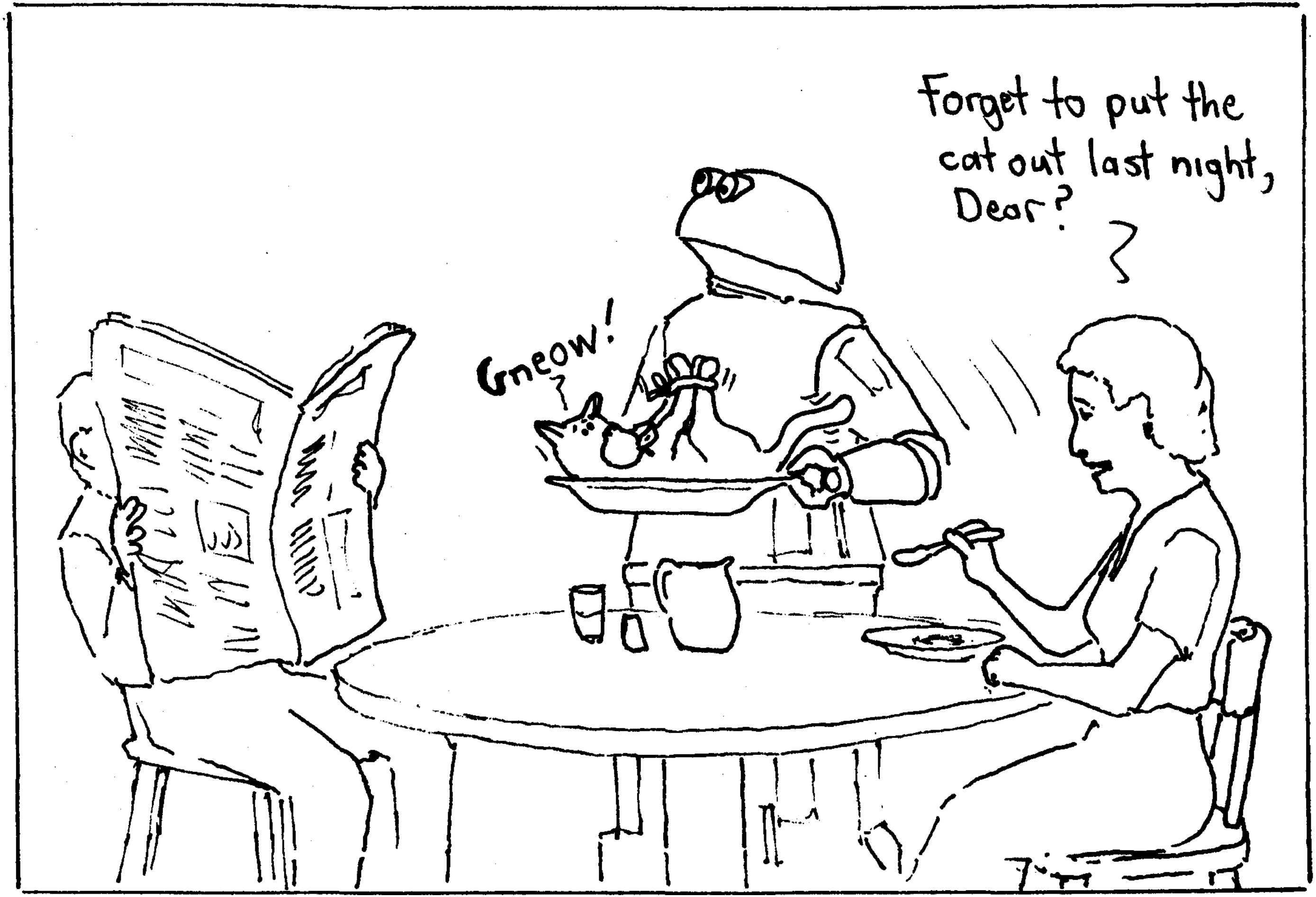


Dinner's ready, Dear

	includes checking its own battery charge etc, and initiates a menu; the first response to that being
ferris	robot trundles off to the kitchen, climbs its ladder on to the bench and readys itself to activate the ferris wheel.
wave	bring microwave into position, and call up programs peculiar to its operation. This time it is a
simple	series of instructions - not your 'cook on medium heat 3 minutes then High for 10 minutes
500	just activate the timer for 5mins 00secs and press the Start 3button.
clock 0635	clever thing knows if it's taken more than 5 minutes it'll adjust this time. I could instead command "wait beep" in which case it would wait until the microwave beeped (Big Ears can tune in to this sound for the beastie)
kettle	turn the ferris to reach the kettle and switch it on
perc	as for kettle - both these routines include a weight test to ensure there is enough water in each before switching them on.
proc 3 20	Bring round the food processor, use speed 3 and give it a 20-second burst. One of these days I'll see if the kids will believe eggshell is really bran flakes!!
toast	We like watching this routine in play. There is a maximum number of slices the beastie can cope with - 20 thick slice, or 29 sandwich slice. This isn't the beastie's fault - it's just that with all the wheeling round, such a high stack is unstable, and the **amxx mice get a real feed from the spinoff. The beastie has a program that makes it find the top of the stack of bread, take the top slice off, put it in slot 1 of the toaster, go back for bread for each of the 3 other slots, set the toaster timer and push the lever down. It waits with its arm over a slot until a piece of toast pops up into it, empties all 4 slots onto a separate plate, goes and gets the next toaster-load, resets the timer down a little... it can cope with a part-load of only 1 slice too.

For beastie operation of the ferris, it helps to keep the appliances in pretty much the same position and angle on their seats eg the frypan thermostat is always on the side nearest the axle, as that is where the beastie has been taught to find the dial.

In fact, with the robot arm hitched up to hubby's computer, family life is undergoing rather a revolution. But more about that another time...


Forget to put the cat out last night, Dear?
Gneow!

ANNUAL GENERAL MEETING

Held 19 March 1986 at the Royal Society Building, Turnbull St, Wellington.

1. APOLOGIES

Don Churchill

2. MINUTES of previous AGM

Having been published in April '85 'Beeblet', they were taken as read.
"That the minutes were a true and accurate record of the meeting" Moved Steve Rose 2nd Jean Heywood. Carried.

3. Matters Arising from Minutes

(a) Canvassing members - in October-November a questionnaire had been sent to all members, results were published in December 'Beeblet', and favoured the status quo.
(b) Paying for contributions or committee members - paying for articles was messy; competitions with a prize incentive had the effect of increasing contributions. The President asked that any further discussion be held to later in the meeting.

4. Correspondence

Nil

5. President's Report

This had been published in February 'Beeblet'
"That the President's Report be taken as read" Moved Neville Hudson 2nd Anton Erasmuson. Carried.
The President then expressed optimism for the club's future now that the Master series was being released at quite a good price in NZ.
He also repeated his thanks to Tim Ryan and Tim Lawson for the work put into the modem project; to Jean Heywood treasurer, Keith O'Callahan equipment and deals, Warren Willis, founding president now standing down from the committee, Anne Flower also founding committee member now standing down, Bruce Wills, editor, and the rest of the committee.

6. Local Group Reports

(a) Nelson:
We have eight Beeb owners in contact with each other to a greater or lesser degree, at least one possibly two are intending to join in the new financial year. At least four have said they would be interested in the VolksModems one has already ordered. Do they have to be members? All the Beeb owners will

have disc systems shortly, seven have printers and all but two have Wordwise /+ or View or both. There is an APTL Sidewise rom/ram , a RAMROM 15, a Solidisk 32K sw.ram, a Watford 32K(MK2) shaddow ram and a Watford(MK2) sw.rom/ram board amongst these eight sets of people. There is also a good range of firmware the most recent one written to work with Acorn DFS , had to have its 40 track accompanying disc changed to work on an 80 track drive ,is the Technomatic NOVOCAD , C A D rom. Maybe the owner will send in a review of that when he has his "eye in". The uses these Beebs are put to are many and varied but all machines are in frequent use. Two of them have recently been down to the Auckland Islands.

In addition there is The BBC/Electron User Group (Nelson) of which you had contact through J.Phelps (incidentally - now moved to Hawkes Bay area and will probably join the National Group from there) . This has only been formed these past six months and has had several pleasant meetings of about twenty people comprising of six of the BBC owners and ten Electron owners. I have tried to recruit at these meetings and have had the Notice re: "half-year-half fee" and Application forms on the Group notice board in Whitcoulls.

(b) Wellington
from Mike Hoult:
Workshops were held at Wellington College, but so far little use had been made of the networking facilities available. The committee tried to meet suggestions of members for activities - a picnic had been held. Warren Willis's Basic course had been a good drawcard. There was a library, with books, software etc; all contributions were always welcome.

7. Financial report

A preliminary report had been presented in February 'Beeblet'; the audited accounts were presented to the meeting. Jean reiterated that the operating surplus came entirely from interest and bulk deals.
A query about stock of old magazines: at the year end they are valued at cost - year just ended at $1; year earlier at 50c; stock of issues prior to this date were written off. New members are sent a list of which issues are still available and their contents.
"That the financial Report be adopted" Moved Jean Heywood 2nd Steve Cosgrove. Carried

9. Moved Anton Erasmuson 2nd Neville Hudson:
That the 1986 subscription for the period 1 April 1986 to 31 March 1987 be set at:

a) annual fee	$30.00
b) student group rate (per person)	$20.00
c) half year fee (1/10/86-31/3/87)	$15.00
d) joining fee	nil

Amendment to motion:
"(e) a rebate of $5 be given to renewing members who pay before 15 May 1986" Moved Jean Heywood 2nd Anton Erasmuson
Some discussion on the budget, which had been drawn up with the rebate in mind. There was $6000 accumulated funds in cash, stock, a printer.
Amendment carried.
Amendment Moved Steve Cosgrove 2nd Steve Rose:
"(f) That the editor of the magazine and committee members have their subscription for the year 1986/7 waived"
All committee members was queried; it was explained that the size of the group was such that the annual cash flow exceeded $30,000; What did 10 committee members do that might warrant waiving fees? The President read out job descriptions covering non-officer committee members for Membership Officer and Equipment officer as examples.
The waiver should be seen not so much as a reward as an inducement for new committee members.
The amendment was put and carried 23-4 on a count of hands.

The amended motion in its entirety was put, Carried unanimously.

10. Moved Keith O'Callahan 2nd John Andrew:
That clause 2(b) of the Constitution be amended by adding to the end of it, the following sentence:
"(The term 'BBC microcomputer' used within this Constitution shall be taken to include all computers marketed by Acorn Computers.)"
Queried whether this should include machines using BBC Basic rather than just Acorn computers; but such machines as Commodore 64 had emulators of the language also.
Carried.

11. Moved Neville Hudson 2nd Keith O'Callahan:
That the heading of clause 3 shall read:
"3. MEMBERSHIP AND ORGANISATION"
Carried

12. Moved Neville Hudson 2nd Keith O'Callahan:
That a clause 3(e) shall be added:
"3(e) With the approval of the Committee, Subgroups may be set up to meet particular needs or interests within the membership

consistent with the objects of The Group, and may finance their activities through a separate levy."
Carried

13. Moved Steve Rose 2nd John Andrew:
That clause 10(b) be amended to read:
"10(b) Proper books of accounts of the Group shall be kept by the Treasurer who shall keep such books up to date. Where required, subgroups shall also keep proper books of accounts and provide a financial statement to the Committee at the end of the financial year."
Carried.

14. Moved John Andersen 2nd Mike Hoult:
That clause 10 (f) be amended to read:
"10(f) All cheques, promissory notes, bills of exchange drawn and endorsements made shall be signed by any two of: the Treasurer, Secretary, and one other committee member nominated by the Committee.
Carried.

15. Election of Officers

Position:	Nominated:	Moved:	Seconded
President	Neil Gordon	Jean Heywood	Anton Erasmuson
Convenor	Glyn Woodbury	Jean Heywood	John Andersen
Secretary	John Andrew	Jean Heywood	Steve Rose
Treasurer	Jean Heywood	Steve H	Glyn Woodbury

These officers were nominated unopposed, and were duly appointed to these positions.
"at least 4 committee members":

1	Keith O'Callahan	Jeff Hunt	Mike Hoult
2	Tim Ryan (in absentia - acceptance to be confirmed)	Steve Rose	Anton Erasmuson
3	Steve Rose	Anne Flower	Anton Erasmuson
4	Neville Hudson	Bruce Wills	Anne Flower
5	Bruce Wills	John Andersen	Steve Rose
6	Steve Cosgrove	Neil Gordon	Anton Erasmuson

Moved Cliff Waldren 2nd Robin Watt:
"That nominations be closed" Carried
Moved Robin Watt 2nd Cliff Waldren:
"That all 6 nominated be elected to the committee" Carried.

16 General Business

(a) Moved Jean Heywood 2nd John Andrew:
"That an honorarium of $2500 be paid to Debbie Levett for making up modems"
This payment had been built into the price for the modems.

* * *

There was a written agreement that the modems were to be built and tested to our specifications, which had P.O. approval. Payment was to be made after work completed not in advanced. Carried.
The availability of modems was explained to newcomers.

(b) Payment for magazine contributions:
Discussion centred on
For solicited articles only: not allowed by Constitution.
Competitions - the committee approves disbursements on the editor's behalf for prizes.
Quality rather than quantity was wanted, so ask for readers' nominations of the 5 best in a period eg 6 months
Monthly prize for best article or program in that issue
Lots of people doing different things, so why should writers only get a reward?

Moved Steve Cosgrove 2nd Jeff Hunt:
"That that committee investigate ways of recognising contributions to the User Group magazine"
Encouragement, not necessarily monetary, was a way of increasing magazine interest.
Carried 20 - 5 on a show of hands

(c) Moved Jean Heywood 2nd Steve Cosgrove that a vote of thanks be given to the auditor Ron Henry. Carried
Next year's auditor needed to be appointed; Ron Henry was provisionally nominated Jean Heywood 2nd Anton Erasmuson. Carried

(d) Moved Neville Hudson 2nd David Fung that a vote of thanks be given to the previous committee. Carried by acclaim.

The meeting closed 9.29pm

* * *

ELECTRON COLUMN

Well, it isn't actually. It is not as if Steve C. did not WANT to provide the column - the arrival of a baby in his household at the critical time drastically reordered his priorities. Mind you, he does seem to have got his hands temporarily on a Master 128 : guess where he was working in May 1983? (Clue : see "Letters...").

* * *

IT TAKES ALL SORTS

OK. So I should be shot. This section contains sort listings. There isn't a lot of documentation with them this month - you will just have to sort them out yourselves.

SHELLSORT AND QUARKSORT

Supplied by Anton E. some time ago. Allegedly Shell Sort is the best general purpose "amateur" sort which will sort anything quickly. For up to 1000 items it is faster than quarksort. Quarksort is interesting in that :
1/ It is an Erasmuson special
2/ "Its called a quarksort because its mainly a quicksort to which is added another sort which carries out while trying to do something which a quicksort needs." Anyway, I think thats what he said.

Both sorts will work with strings, integers or reals if the array variables are changed appropriately. This is most easily done with Search and replace on a wordprocessor.

Persons interested in more details are invited to write to Anton care of the UG box.

GENERAL PURPOSE SORTING PROGRAM

I have had this sort of sitting here for a while too. Now that he knows that its going to be published, Steve's plans to document it have been brought forward a little... next month? I understand it has a few interesting features...

```
10 REM  Quarksort and Shellsort
20 REM Two sorting PROCedures.
30 REM Anton Er. Sept,'85
40
50 MODE6:@%=&80A
60 DIM A$(1000)
70 REM A$(0) used in swapping
80 REM Arrays of 5 to 3000 items
90 REM  are sorted and timed.
100 FOR scan=1.705 TO 6.906STEP .1
110   N%=EXP(scan)
120   PROCrandom:REM Fake data
130   T%=TIME
140   REM PROCquark(1,N%)
141   PROCshell(N%)
150   T%=TIME-T%
```

```
 160   PROCoutput
 170   NEXT
 180 END
 190
 200 DEF PROCrandom
 210 FOR I%=1 TO N%
 220   T$=CHR$(64+RND(26))
 230   FOR J%=1 TO 5
 240     T$=T$+CHR$(96+RND(26))
 250     NEXT
 260   A$(I%)=T$
 270   NEXT
 280 ENDPROC
 290
 300 DEF PROCoutput
 310 FOR I%=1 TO N%
 320   PRINTTAB(I%MOD4*10) A$(I%);
 330   NEXT
 340 PRINT' N%;" Items take "T%/100;" s
    ecs."''
 350 ENDPROC
1999
2000 DEF PROCquark(P%,Q%)
2002 LOCAL I%,J%,K%
2004 IF P%>=Q% THEN ENDPROC
2006 K%=FNpivot(P%,Q%)
2008 IF K%<0 THEN ENDPROC
2010 I%=P%-1:J%=Q%+1
2012 REPEAT
2014   REPEAT:I%=I%+1:UNTIL I%=J%-1 OR
       A$(I%)>A$(K%)
2016   REPEAT:J%=J%-1:UNTIL I%=J% OR A$
       (J%)<=A$(K%)
2018   PROCswap(I%,J%)
2020   UNTIL I%>=J%-1
2022 IF A$(I%)>A$(K%) THEN I%=I%-1
2024 PROCquark(P%,I%)
2026 PROCquark(I%+1,Q%)
2028 ENDPROC
2030
2032 DEF FNpivot(P%,Q%):REM Quark FN
2034 IF Q%>P%+4 THEN PROCsample
2036 I%=P%:J%=P%
2038 REPEAT:I%=I%+1:UNTIL I%=Q% OR A$(P
    %)<>A$(I%)
2040 IF A$(I%)<A$(P%) THEN PROCswap(I%,
    P%) ELSE J%=I%-1
2042 IF I%=Q% THEN =-1
2044 REPEAT
2046   J%=J%+1
2048   REPEAT:I%=I%+1:UNTIL I%=Q% OR A$
       (J%)<>A$(I%)
2050   IF A$(I%)<A$(J%) THEN PROCswap(I
       %,J%)
2052   UNTIL I%=Q% OR A$(P%)<>A$(J%)
2054 IF A$(J%)<A$(P%) THEN PROCswap(P%,
    J%)
2056 IF I%=Q% THEN =-1
2058 =J%
2060
2062 DEF PROCsample:REM Quark PROC
2064 J%=1+Q%-P%
2066 FOR I%=P% TO P%+2
2068   PROCswap(I%,RND(J%)+P%-1)
2070   NEXT
2072 ENDPROC
2074
2076 DEF PROCswap(X%,Y%):REM Quark PROC
2078 IF X%=Y% THEN ENDPROC
2080 A$(0)=A$(X%):A$(X%)=A$(Y%):A$(Y%)=
    A$(0)
2082 ENDPROC
2999
3000 DEF PROCshell(N%)
3002 IF N%<2 THEN ENDPROC
3004 LOCAL I%,J%,K%
3006 I%=&1FFF
3008 REPEAT:I%=I%DIV2:UNTIL I%<N%
3010 REPEAT
3012   FOR J% = I%+1 TO N%
3014     IF A$(J%)<A$(J%-I%) THEN PROCi
       nsert
3016     NEXT
3018   I%=I%DIV2
3020   UNTIL I%<1
3022 ENDPROC
3024
3026 DEF PROCinsert:REM Shell PROC
3028 A$(0)=A$(J%)
3030 K%=J%
3032 REPEAT
3034   A$(K%)=A$(K%-I%)
3036   K%=K%-I%
3038   UNTIL FNfnd
3040 A$(K%)=A$(0)
3042 ENDPROC
3044
3046 DEF FNfnd:REM Shell FN
3048 IF K%<=I% THEN =TRUE
3050 IF A$(K%-I%)>A$(0)THEN =FALSE ELSE
     =TRUE
 20 REM General Purpose sorting progra
    m
 30 REM written by Steve Cosgrove, Jul
    y 1985
 40 REM version 1.1, September 1985
 50 MODE 6
 60 PROCinit
 65 PROCtest_data
 70 PROCinput
 80 PROCsort(sort_byte%,datalines%-fir
    stsort%)
 90 PROCsave
100 PROCtidyup
110 END
120
130 DEF PROCinit
140 @% = &00608   :  REM change the fi
    eld length of numbers in PRINTs
150 maxchar% = 127 : REM maximum numbe
    r of charactors per line of data
160 eoln% = 13   :   REM End Of LiNe c
    harcter
170 eof% = 0     :     REM in VIEW the f
    ile is terminated with  0D 00
180 namelength% = 11:REM maximum lengt
    h of filename for cassette base
190 suffix$="srt"  : REM added to what
     ever filename is entered in PROCs
    ave
200 datalines% = 0 : REM total number
    of lines input
210 firstsort% = 1 : REM first number
    to include in sort. Usually >1 for
     VIEW
220 first_byte%=&F00:REM location of s
    tart of VIEW file
230 sort_byte% = first_byte% : REM loc
    ation of start of line "firstsort"
240 last_byte% = first_byte% : REM end
     of file marker, reset by PROCinpu
    t
250 DIM temp% maxchar%    :    REM spa
    ce to be used in PROCswap
260 invalid$ = ""   : REM charactors wh
    ich should not be in a filename
270 FOR I% = 0 TO ASC(" ")
280   IF I% <> 13 THEN invalid$ = inva
      lid$ + CHR$(I%)
290   REM RETURN is ok
300   NEXT
310 invalid$ = invalid$ + CHR$(34) : R
    EM '"' is invalid
320 ENDPROC
323
330 DEF PROCtest_data
340 P% = first_byte%
350 FOR I% = 1 TO 20
360   L% = RND(20)
370   FOR J% = 1 TO L%
380     ?P% = RND(100)+32
390     P% = P% + 1
400     NEXT
410   ?P% = eoln%
415   P% = P% + 1
420   NEXT
430 ?P% = eof%
440 ENDPROC
450
460 DEF PROCinput
470 LOCAL pointer%,started%
480 REM this proceedure checks how muc
    h valid data can be found
490 REM it does not process it at all
495 pointer% = first_byte%
500 REPEAT
510   pointer% = pointer% + LEN($point
      er%) + 1
520   REM begining of next line
530   IF FNformatline AND NOTstarted%
      THEN PROCdont_start ELSE started
      % = TRUE
540   REM this line is to avoid trying
       to sort VIEW format commands
550   datalines% = datalines%+1
560   PRINT ;datalines%;" lines found.
      "; CHR$(13);
570   last_byte% = pointer%
580   REM last byte% points to the beg
      inning of next line, or EOF chr.
590   UNTIL ?pointer% = eof%
600 PRINT '"They are in memory from ";
    ~first_byte%;" to ";~last_byte%'"T
    he first ";firstsort%;" are format
    ing lines."
610 REM location of all lines of data
    to be sorted
620 ENDPROC
630
640 DEF FNformatline
650 REM In VIEW, lines used for format
    ting begin with a chr > 127
```

```
660 = (( LEFT$($pointer%,1) )= CHR$(12
    7)) OR LEN($pointer%) = 0)
670
680 DEF PROCdont_start
690 firstsort% = firstsort% + 1
700 sort_byte% = pointer% + LEN($point
    er%) + 1
710 REM Line number, and location, of
    current first line to sort
720 ENDPROC
730
740 DEF PROCsort(bottom%,top%)
750
760 REM This is a bubble sort.
770 REM it is bloody slow, but does no
    t disrupt the contents
780 REM of memory too much, as it only
     ever swaps a line with
790 REM the line next to it
800 REM PS - thats also why it's so sl
    ow
810
820 LOCAL pointer_1%,pointer_2%,swap%
830 REM pointers to each of the string
    s being compard
840 REM swap checks whether or not lis
    t is sorted yet
850 REM ie, if nothing is changed in a
     complete pass, sort is complete
860 PRINT '"Now sorting"'"the counter
    below shows"'"how many loops remai
    n."
870 REPEAT
880   pointer_1% = bottom%
890   swap% = FALSE
900   count% = 1
910   PRINT CHR$(13); top% - count% "
      + " top% "    ";
920   REPEAT
930     pointer_2% = pointer_1%+ LEN($
        pointer_1%)+1
940     IF FNdo_swap THEN PROCswap
950     pointer_1% = pointer_2%
960     count% = count%+1
970     UNTIL count%>=top%
980   top% = top%-1
990   UNTIL (NOT swap%)
1000 CLS
1010 PRINT '"Sort complete"
1020 ENDPROC
1030
1040 DEF FNdo_swap
1050 REM this Function could be modifie
     d to sort on a specific key
1060 = $pointer_2%<$pointer_1%
1070
1080 DEF PROCswap
1090 REM this routine swaps two adjacen
     t strings in memory
1100 swap% = TRUE       :        REM An
      element has been swapped
1110 $temp% = $pointer_1%    :    REM sa
     ve the first item
1120 $pointer_1% = $pointer_2% : REM ov
     erwrite the first line with second
1130 pointer_2% = pointer_1% + LEN($poi
     nter_1%) + 1 : REM reposition poin
     ter 2
1140 $pointer_2% = $temp%    :    REM pu
     t the first line into its new plac
     e
1150 ENDPROC
1160
1170 DEF PROCsave
1180 LOCAL exit%,saved%,vpos%
1190 *opt1,1
1200 saved% = FALSE : REM haven't saved
      the file yet!
1210 REPEAT
1220   filename$ = FNfilename(NOT saved
       %,filename$)
1225   exit% = (filename$ = "")
1230   IF saved% AND exit% THEN PRINT '
       "Saving again"
1240   IF (NOT exit%) THEN OSCLI ("save
        "+filename$+" "+ STR$~(first_by
       te%)+" "+ STR$~(last_byte%)) : s
       aved% = TRUE
1250   *opt1,0
1260   REM suppress messages for second
        and subsequent copies
1270   IF exit% AND (NOT saved%) THEN P
       RINT '"File not saved" ELSE IF (
       NOT exit%) THEN PRINT "File save
       d"'
1280   UNTIL exit%
1290 ENDPROC
1300
1310 DEF FNfilename(first_pass%,current
     $)
1320 LOCAL exit%, return%, first_letter
     $
1330 error%=FALSE
1340 REPEAT
1350   PRINT "Hit X to exit without sav
       ing."
```

```
1360   IF current$ ) "" THEN PRINT "[illegible]
        return to leave filename unchan
       ged,"'"(it is currently "current
       $"),"'"  and save again."
1370   PRINT "Otherwise enter a ";
1380   IF (NOT first_pass%) THEN PRINT
       "new ";
1390   PRINT "filename."
1400   IF first_pass% THEN PRINT "You w
       ill be prompted to put "'"the ta
       pe recorder onto record,"
1410   IF first_pass% OR error% THEN PR
       INT "The filename must be less"'
       "than ";namelength%" charactors
       long."'"The suffix '."suffix$"'
       will be added"'"  if appropriate
       ."
1420   first_letter$ = GET$
1430   return% = (first_letter$ = CHR$(
       13))
1440   exit% = (first_letter$ = "X") OR
        (first_letter$ = "x") OR return
       %
1450   IF exit% AND (NOT return%) THEN
       current$ = ""
1460   IF (NOT exit%) THEN current$ = F
       Ninput(first_letter$,invalid$) :
        PROCtidy_name
1470   REM if the first letter is an "X
       ", "x", or RETURN, this line is
       omitted
1480   REM otherwise it is totally tran
       sperant to the user
1490   IF (LEN(current$) )= namelength%
       ) THEN error% = TRUE : current$
       = "" : VDU 7
1500   IF (current$ = "") AND return% T
       HEN error% = TRUE
1510   UNTIL (NOT error%) OR exit%
1520 = current$
1530
1540 DEF FNinput(current$,invalid$)
1550 REM first charactor(s), list of in
     valid charactors for input
1560 LOCAL result$, new$, I%, flag%
1570 REM string to be output, each char
     actor read, counter for loop
1580 result$ = ""
1590 FOR I% = 0 TO LEN(current$)
1600   IF I% ) 0 THEN flag% = INSTR( in
       valid$, MID$(current$,I%,1))
1610   IF (I% ) 0) AND flag% THEN VDU 7
        : ELSE IF I% ) 0 THEN PRINT MID
↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑$(
       $(current$,I%,1); : result$ = re
       sult$ + MID$(current$,I%,1)
1620   REM using 0 accounts for the pos
       ibility of current being ""
1630   REM Otherwise print, and include
        in result, any valid charactors
1640   NEXT
1650
1660 REPEAT
1670   new$ = GET$
1680   IF INSTR( invalid$,new$ ) THEN V
       DU 7  ELSE PRINT new$; : IF new$
        = CHR$(127) THEN result$ = LEFT
       $(result$, LEN(result$)-1) ELSE
       IF new$ () CHR$(13) THEN result$
        = result$ + new$
1690   UNTIL new$ = CHR$(13)
1700 VDU 10 : REM line feed
1710
1720 = result$
1730
1740 DEF PROCtidy_name
1750 REM string to be output, each char
     actor read
1760 IF (INSTR(current$,".") = 0) AND (
     (LEN(current$) + 1 + LEN(suffix$))
      (= namelength%) THEN current$ = c
     urrent$ + "." + suffix$
1770 FOR I% = 0 TO LEN(current$)
1780   REM assumes a suffix will be sep
       erated by a dot.
1790   REM adds suffix if one does not
       already exist,
1800   REM and it does not make the nam
       e too long.
1810   ENDPROC
1820
1830   DEF FNsyntax_ok(name$)
1840   LOCAL position%
1850   FOR check% = 1 TO LEN(invalid$)
1860     position% = INSTR(name$,MID$(i
         nvalid$,check%,1))
1870     IF position% ) 0 THEN PRINT "C
         haractor number "position%" is
          invalid."
1880     NEXT
1890   = (position% = 0) : REM ie - no
       invalid charactors in name
1900
1910   DEF PROCtidyup
1920   *opt1,1
1930   PRINT STRING$(4, CHR$(10));"E[illegible]
        SORT program."
1940   ENDPROC↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑
```

ADVANCED DISC FILING SYSTEM II

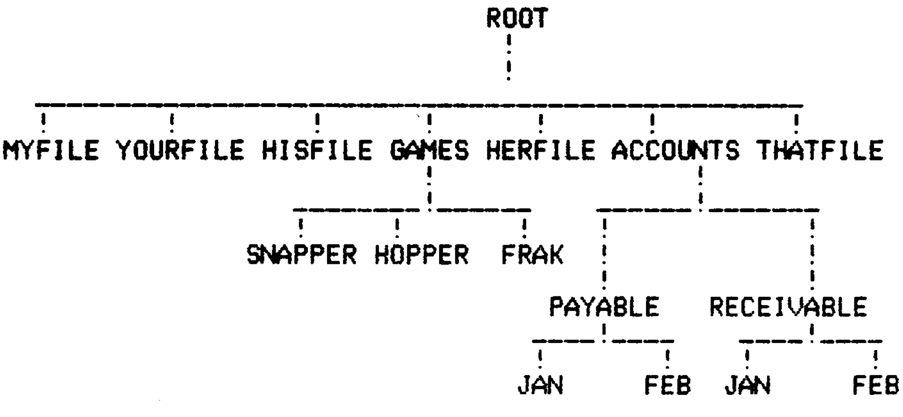
Hierarchical Directories by Tony Krzyzewski

The concept behind that rather horrendous name (by the way, it's pronounced HIGHER-ARE-KICK-AL) is fairly simple. Each file stored on disc is given a name in order to make it unique. In order to keep track of lots of filenames we organise them into a directory, rather like the contents of a book. For example

THISFILE
THATFILE
MYFILE
YOURFILE
HISFILE
HERFILE

The ADFS also stores information concerning whereabouts on disc the information is stored and where in memory it came from. Normally you can't see this information as it is hidden, but later on we will look at commands to let us see this information. With only a few files in the directory it is easy to see what files are available. Once we get a large number of files it becomes rather cumbersome. It is because of this that sub-directories were introduced. Instead of a directory only being able to contain files, it may also contain directories. These sub_directories may in turn contain more files and directories. In order to distinguish the directory we started, from all the other directories, we call that directory the ROOT.

Let's look at a directory structure:



In this example, the ROOT directory contains the following files:
MYFILE
YOURFILE
HISFILE
HERFILE
THATFILE

and also contains the sub_directories
GAMES
ACCOUNTS

The games directory in turn contains 3 files
SNAPPER
HOPPER
FRAK

whilst the accounts directory contains 2 sub_directories
PAYABLE and RECEIVABLE
but doesn't contain any files.

Both the PAYABLE and RECEIVABLE sub-directories contain 2 files
JAN and FEB

It is here that you will note that a filename does not have to be unique - IF, and ONLY IF, the two files are contained in SEPARATE sub-directories.

A few questions come to mind here:
How do we know which FEB file we are talking about, and how do we tell the sub-directories from the files?

Before these questions can be answered, we have to define some naming conventions:
1. An OBJECT is a file name or a directory name
2. A DIRECTORY is a collection of objects (up to 47)
3. A FILE is a block of data or a program that can be loaded into the computer. For the sake of these examples we shall treat all files as LOADable BASIC programs.
4. A file name or directory name may be up to 10 characters long and the name shall not contain any of the following characters:
* . : $ & ^ nor space
5. The ROOT directory is referred to as $
6. The directory currently selected is known as the CURRENTLY SELECTED DIRECTORY (CSD) and may be referred to as @
7. The directory immediately preceding the CSD in the structure

is that directories PARENT and is referred to as ^

With that lot behind us, some examples.

If we were currently in the ACCOUNTS directory (next month shows how to get there!):
The CSD would be ACCOUNTS
The CSD contains two objects PAYABLE and RECEIVABLE
The PARENT directory is $

Next month we answer those two questions, but before then, here are some for you.

1. How many sectors are there on a double-sided, 80 track disc?
2. What is the Parent directory of GAMES?
3. What is the Parent directory of PAYABLE?
4. In what sub_directory is the SNAPPER program?

PLINTH

For the BBC Microcomputer

CTI: "...LISP-like?"

PCWW: "...allows greatly-increased user-friendliness"

Core Users Weekly: "...highly structured..."

Biblet: "Provides a new standard to harness and combine Beeb capabilities with those of modern screens"

No extra cables or soldering needed;
Ergonomically designed - will not handicap further expansion boards - slots allow other cables etc free movement to scarce inside-Beeb spaces.

Full documentation provided; includes full Assembly Instruction Set

NZ Agents: Bizley Ohms Ltd
Private Bag
Kaingaroa

send $55.95
24-hour despatch.

CIVILISED MODE CHANGE

The Centaur, Hamilton

Editor's note: this was contribution was sent in last year and I have finally managed to type it in (disc or tape material alwys gets precedence...).

Well, after being made to feel extremely guilty after reading the latest Beeblet I have decided to make my presence known. I have been using a BBC micro for two years (almost non-stop) and have a reasonable knowledge of what makes it tick. When (if) I contribute programs, hints etc. to the Beeblet they will mostly be specific to the BBC micro, and probably not work on the Electon (a machine I am not familiar with so that if a routine works on it, it wasn't my fault). I will also try in future to answer any HELP queries anyone may have on the condition that they relate to the Beeb.

Just to whet everyone's appetites, I have included a routine which will allow you to change screen mode without clearing the screen. Why would you want to change screen mode without clearing the screen? Well take the example of a game that loads a HiRez, complicated background. Off tape this can take up to 15 minutes. Now suppose the game switches to mode 7 to display the HiRez scoreboard. Upon returning to the Hirez screen normally the picture would be lost, but with this routine you only lose the parttaken up by the Mode 7 screen. If necessary a copy of that section of the screen could be stored elsewhere in the machine and reloaded each time.

This routine uses a lot of unknown Op System calls and so will probably only work on O.S. 1.2. I realise that this makes me only of limited use, but when you start writing fast M/C routines they become machine specific. To use the routine just make A%= (mode required): and then CALL (location code was assembled). A quick rundown of what it does:

Lines	Description
40	Saves accumulator which contains mode number
50-110	Zero all the O.S. VDU variables (X,Y,coords etc)
120	Reset the character font locations to default

130-410	Reset mode dependant VDU variables
420-450	Set System VIA for hardware scrolling
460-610	Restore general VDU variables
620-630	Set VDU status as read by OSBYTE &75
640-660	Reset flashin colour timers
670-770	This is the section that actually changes the screen mode by altering the CRTC registers
780-790	Reset text and graphic colours etc.
800-960	Restore last of the VDU variables, pointers to lookup tables etc.

And if all that hasn't completely flummoxed you, then let me close by saying that once this code has been assembled, it can be saved to disk by typing *SAVE filename startaddress +D6

The code has been written so that it is completely relocatable (unlike socks) and so that it can be re-*LOADed anywhere in memory.

```
 10 REM MODE CHANGE PROGRAM
 20 REM BY THE CENTAUR
 30
 40 FOR T%=0 TO 2 STEP 2
 50   P%=&900
 60   [OPT T%
 70             PHA
 80             LDX #&7F
 90             LDA #&00
100 .LOOP1
110             STA &D0
120             STA &02FF,X
130             DEX
140             BNE LOOP1
150             JSR &CD07
160             PLA
170             LDX #&7F
180             STX &0366
190             BIT &028E
200             BMI FWRD1
210             ORA #&04
220 .FWRD1
230             AND #&07
240             TAX
250             STX &0355
260             LDA &C414,X
270             STA &0360
280             LDA &C3FF,X
290             STA &034F
300             LDA &C43A,X
310             STA &0361
320             BNE FWRD2
330             LDA #&07
340 .FWRD2
350             ASL A
360             TAY
370             LDA &C406,Y
380             STA &0363
390 .LOOP2
400             ASL A
410             BPL LOOP2
420             STA &0362
430             LDY &C440,X
440             STY &0356
450             LDA &C44F,Y
460             JSR &E9F8
470             LDA &C44B,Y
480             JSR &E9F8
490             LDA &C459,Y
500             STA &0354
510             LDA &C45E,Y
520             STA &034E
530             TYA
540             ADC #&02
550             EOR #&07
560             LSR A
570             TAX
580             LDA &C466,X
590             STA &E0
600             LDA #&C3
610             STA &E1
620             LDA &C463,X
630             STA &0352
640             STX &0353
650             LDA #&43
660             JSR &C5A8
670             LDX &0355
680             LDA &C3F7,X
690             JSR &EA00
700             PHP
710             SEI
720             LDX &C469,Y
730             LDY #&0B
740 .LOOP3
750             LDA &C46E,X
760             JSR &C95E
770             DEX
780             DEY
790             BPL LOOP3
800             PLP
810             JSR &C839
820             JSR &C9BD
830             LDX #&00
840             LDA &034E
850             STX &0350
860             STA &0351
870        JSR &C9F6      950        STY &035E
880        LDY #&0C       960        LDX #&00
890        JSR &CA2B      970        STX &0269
900        LDA &0358      980        STX &0318
910        LDX &0356      990        STX &0319
920        LDY &C454,X   1000        RTS
930        STY &035D     1010        ]
940        LDY #&CC      1020   NEXT
```

BUDGET MEMORY EXPANDER

Unknown

On page 237 of my User Guide it explais DIMensioning of arrays and also how to reserve bytes in memory for special applications. An example is given and also a practical illustration on page 463. Turning the cassette motor light on does not appeal to me as the ideal illustration especially as I have to use BREAK to turn it off. A routine to change the DIRectory on a disc would be

```
10 DIM C 4
20 $C="DIR A"
30 X%=C MOD 256
40 Y%=C DIV 256
50 CALL &FFF7
```

and RUN. hoops I didn't reserve enough space and it worked. But 0 counts as well so its alright. But what if it was not enough... lets make the 4 a 0.. it ran again.. well, how about -4.. did it work.. backward RAM.. So all you need is some method of putting all the programs into DIM statements and the lack of memory is solved.

CLASSIFIED ADS

(no charge for members)

FOR SALE:

H.C.R. Sideways RAM board, write protectable, easily fitted, $75

Two 6264LP-8K RAM chips, new, $20

Phone E. Wong, Palmerston North, 699-72 (evenings)

LETTERS.....

David Fung

It was in May 1983 upstairs at the then PROJECT COMPUTER that I came across a bunch of amiable but somewhat off beat characters. With their help and encouragement I have wrestled with the many problems of a long distance programmer in the development of a Chinese Word Processor on the Beeb in these three years. By May 1986 the project will be completed if Acorn does not spring on me another new machine.

Some of the user group members and some non members had made enquiries or had written to me about my project. I have been very remissed in responding to them and I offer my sincere appologies. My excuse is that I desperate wanted to put 48 hours into a 24 hour-day. In the last year or so I have literally hand-crafted over 4300 chinese characters and entered them into data files. The time for this alone has been well over 2000 hours.

The problem of constantly access the disc files for data meant that the operation was slow (even if it was faster then other systems), and that the clicking on and off was very hard on the disc drives. The rest of my time was spent in hassling David Harris to get me the Solidisk SWR 256, which was never produced until now. Eventually the SWR 128 came which partially helped with less then 100 KB ramdisc. This was achieved amidst buggy software, including several modifications, and upgrades of the DDFS Rom. Came January 1986 (November in U.K - Time warp!) OPUS came up with the Challange 3 disc drive with 2 x 256 K Ramdiscs. This was pivotal in the whole development, with the Challanger accessing of the data is now instantaneous and blissfully silent.
The programme had to be modified of course. This was about done, and I wrote to Joe Joyce of Barson about it. He said " Come to have a look at the Master 128 ". I did, now I am rewriting the programme once more. I will take a while longer but will be worth it because with the Master 128 I would not need the 6502 second Processor, without loss of speed, and, actually less expensive to configure the hardware.

This then is the news up to the moment, next time will be the real thing. Many thanks to every body with their interest and assistance, and especially your own encouragement.

HALL OF FAME

PUBLISHER:	GAME	HISCORE	SCORER
Aardvark	Frak	1,026,820	Andrew P.
	Monsters	175,110	Geof P.
	Zalaga	590,100	Allister J.
		52,090*	Nick C.
Acorn	Arcadians	34,740	Rasik R.
	Aviator	5,610	Joanne C's Dad
	Countdown	246	Andrew
	Meteors	46,200	Andrew P.
	Monsters	149,730	Rupert T.
	Planetoid	413,875	Peter J.
	Rocket Raid	140,777	Mark R.
	Snapper	250,000	Susan
	Starship Command	9,495	Robert N.
	Super Invader	59,250	Kiran R.
A&F	Cylon Attack	157,970	Aaron
	Chuckie Egg	319,450*	Greg C.
Alligat.	Bug Blaster	37,642	Steve W.
Amcom	Fortress	89,900	Andrew P.
Bugbyte	Twin Kingdom	385	A. Beeston
Comp.Con	Android Attack	236,195	Peter J.
Dr Soft	'747'	205	Steve W.
Firebird	Bird Strike	11,340	David G.
		14,980*	Patrick C.
FG Soft.	Mutant Defender	6,900	Steve W.
	Lords of Time	925/1000	Steve W/Kerry K.
	Snowball	900/1000	Steve W./Dianne
Micro. Pow	Castlequest	50	Jeremy B.
Prog Pow	Felix in Factory	13,150	Duncan's Flatmate
	Frenzy	915,468	Dianne T.
	Killer Gorilla	91,000	Susan
	Laser Command	109,225	P. Leadley
	Moon Raider	112,350	P.Leadley
Sup.Soft	Q*Bert	43,825	Peter J.
	Mr Wiz	21,050*	Chris M.
	Roadrunner	118,900	Duncan's Flatmate
?	Dr Who	35,240	Duncan's Flatmate

*Electron Version.

HALL OF FAME

USER GROUP MEETINGS

AUCKLAND: -meets 2nd Wednesday of the month, 7.30pm, Conference Rooms, Auckland Secondary Teachers College,60 Epsom Ave, Epsom. ph Dave 770-630 x 518(bus) or Kerry 695-355.
-North Shore: meets last Wednesday of the month, 7.30pm, Salvation Army Hall, Glenfield Road, Glenfield ph Chris 419-0543 (wk)
CHRISTCHURCH: -fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.
DUNEDIN: -first Thursday of the month. Phone Martin McDowall 42-031
HAWKES BAY: -Alternative months Hastings & Napier. Contact Kendall Napier 435-624, Bob Taradale 446-955, Mitch Hastings 778-235.
INVERCARGILL: -Phone Evan 330-444 Invercargill.
PALMERSTON NORTH: -Phone Tom Skinner 83-009(wk)
TAUPO: -meets every second Wednesday @ 7pm. Contact David 84-215 (h), 85-124 (wk).
TAURANGA: -every second Wednesday. ph. Chris & Jane 65-076.
TIMARU: -Phone Lloyd van der Krogt Timaru 61-412.
TOKOROA: -meets first Friday of each month, Tokoroa High School, 8pm. Phone Brian 67-625 Tokoroa.
WAIKATO: -first Friday of the month 7.30pm Waikato Tech Institute. Ph. Alison or John, Morrinsville 6695.
WELLINGTON: -meets usually second WEDNESDAY of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon. Other dates advised in Beeblet regional news.
-Workshops: fourth Wednesday of month, other venues. ph. Don 848-235 or Beryl 862-874 or Mike 785-437.
OTHER CENTRES: - let us know and we will publish details here.

THE NEWSLETTER:

CONTRIBUTIONS:-most welcome. Listings should be sent in on tape or disc. (Tape - 2 copies, one at 300 baud please). Please include written explanation of listing; text and graphics only material welcome too.

DEADLINES:- For a particular issue, the last day for material is the last day of the month prior to publication.

ADVERTISING:- Rates are $20 per half page (camera ready), deadline as above.

Copyright (c) **BBC/ACORN COMPUTER USER GROUP OF NZ INC.**

Published monthly except January and mailed to financial members.

MEMBERSHIP:

Membership of the users group is on payment of an annual subscription from April to March. For the 1985/6 year it is: $30.00, plus $5.00 for Auckland & Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Joining or renewing members get all the back issues of the current year - those joining from 1 October may have half the annual sub waived and only receive newsletters from 1 October.

THE BBC/ACORN COMPUTER USER GROUP OF NZ INC, P.O.Box 9592, WELLINGTON.

Reproduced by BRYCE FRANCIS LTD ,WELLINGTON